Army Research Laboratory

ScrapRE

by Christine E. Slocum

ARL-TN-383 January 2010

Army Research Laboratory
Aberdeen Proving Ground, MD 21005-5067

ARL-TN-383 **January 2010**

ScrapRE

Christine E. Slocum
Computational and Information Sciences Directorate, ARL

REPORT DOCUMENTATION PAGE			*Form Approved* *OMB No. 0704-0188*
Public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden, to Department of Defense, Washington Headquarters Services, Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number. **PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.**			
1. REPORT DATE *(DD-MM-YYYY)* January 2010	**2. REPORT TYPE** Final		**3. DATES COVERED (From - To)** May 2007–May 2008
4. TITLE AND SUBTITLE ScrapRE			**5a. CONTRACT NUMBER** W81XWH-05-C-0058
			5b. GRANT NUMBER
			5c. PROGRAM ELEMENT NUMBER
6. AUTHOR(S) Christine E. Slocum			**5d. PROJECT NUMBER**
			5e. TASK NUMBER
			5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army Research Laboratory ATTN: RDRL-CII-C Aberdeen Proving Ground, MD 21005-5067			**8. PERFORMING ORGANIZATION REPORT NUMBER** ARL-TN-383
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)			**10. SPONSOR/MONITOR'S ACRONYM(S)**
			11. SPONSOR/MONITOR'S REPORT NUMBER(S)
12. DISTRIBUTION/AVAILABILITY STATEMENT Approved for public release; distribution is unlimited.			
13. SUPPLEMENTARY NOTES			
14. ABSTRACT The U.S. Army Research Laboratory's Automatic Construction of Urgently-needed Translation Engines (ACUTE) initiative develops foreign-language machine translation (MT) systems. Development conducted July through August 2007 in support of ACUTE MT efforts produced ScrapRE, a textual entity extractor for online news articles. ScrapRE has been used since September 2007 to compile an Urdu-language document corpus.			
15. SUBJECT TERMS ACUTE, Urdu, information extraction, regular expression, HTML			

16. SECURITY CLASSIFICATION OF:			**17. LIMITATION OF ABSTRACT**	**18. NUMBER OF PAGES**	**19a. NAME OF RESPONSIBLE PERSON** Christine E. Slocum
a. REPORT Unclassified	**b. ABSTRACT** Unclassified	**c. THIS PAGE** Unclassified	UU	16	**19b. TELEPHONE NUMBER** (*Include area code*) 410-278-4958

Standard Form 298 (Rev. 8/98)
Prescribed by ANSI Std. Z39.18

Contents

List of Figures

1. Introduction

The Automatic Construction of Urgently-needed Translation Engines (ACUTE) initiative of the U.S. Army Research Laboratory (ARL) develops foreign-language machine translation (MT) systems. The remainder of this section describes Urdu corpora needed for ACUTE language modeling and introduces ScrapRE, a textual entity extractor developed to compile the corpora from online news articles. Section two details ScrapRE implementation. ScrapRE system requirements, configuration, usage, and output are documented in section three.

A natural-language corpus is a collection of uniformly annotated text documents of a single language, written by fluent speakers; collectively, corpora exhibit the language's unique semantic and syntactical features. MT engines that employ Statistical Language Modeling (SLM) techniques capture these features "by estimating the probability distribution of various linguistic [entities], such as words, sentences, and whole documents. "[1] To train a statistically significant language model, a large corpus must be available that contains document entities structured and annotated per requisite of the MT engine's schema. However, document sources rarely originate in structured form, so Information Extraction (IE) techniques are employed to search for and annotate document features associated with schema-declared entities.

Current ACUTE development of an Urdu language model requires a large Urdu text corpus. International news websites featuring Urdu articles have been identified as rich, ever-growing sources of the needed text, but article entities are embedded in unstructured HTML source files and are not easily accessible. A textual entity extractor, ScrapRE, was developed during July and August 2007 to extract entities from online BBC and VOA Urdu news articles.

ScrapRE relies on the dynamic page generation process employed by the news sites—unique database content is inserted into templates, creating article sources that exhibit the template's features. Through reverse engineering, common markup features can be identified among samples of sources generated from the same templates, then interspersing regions of unique content can be mapped to schema-named entities in *wrappers*.[*] ScrapRE interfaces article sources with site-specific wrappers to mask template regions and expose named article entities for extraction.

Textual entity wrappers were manually encoded for the BBC and VOA Urdu news articles by analyzing the HTML source code of sample articles. The unique textual content in each sample was captured from an Internet browser's rendition of the article, then searched for and highlighted directly in the sample's source with entity name annotations. Non-highlighted

[1] Rosenfeld, R. Two Decades of Statistical Language: Where Do We Go From Here? *Proceedings of the IEE*, August 2000; Vol. 88, No. 8, pp 1270–1278.

[*] The set of regular expression patterns for a particular domain is known as the domain's "wrapper," when the patterns are used collectively to interface and extract entities from a source.

source code regions from all samples were then compared, revealing template patterns associated with specific entities as well as irregular markup not attributed to templates or content. The identified template patterns were generalized and merged when possible to form a set of wrappers for the article sources.

2. Implementation

ScrapRE is given as input a collection of links referencing either article webpages or RSS feeds, depending on program settings. If provided with article links, ScrapRE fetches the article's HTML source files directly; if provided with feed links, the program must parse article links embedded in the feeds, then fetch these article's sources. All sources retrieved from a domain are grouped together and persisted locally for further processing.

A corresponding set of regular expression (RE) patterns must exist for each domain represented among the persisted source files; the patterns collectively form a wrapper to interface domain articles by mapping the sought-after named entities to all identified variants of the HTML code in which they might be interspersed. ScrapRE iterates over each domain once during execution, loading the domain's corresponding pattern files into memory and compiling the patterns as binary objects accessible to a regular expression engine. The RE engine then scans for pattern matches in all domain sources, following the customized search routine outlined in figure 1.

```
Place all patterns in a (first-in-first-out) queue
Create empty dictionary* matches
REPEAT
  Let index marker flag = 0
  Pop pattern from queue
  WHILE flag < len-1
    IF flag falls within an already matched index range THEN
      Set flag at the end of the range
    END IF
    Search for match to pattern in src, starting from flag
    IF match found THEN
      Insert region into matches for the respective named-entity key
      Record indices of matched characters
      Set flag at last matched index
    END IF
  END WHILE
UNTIL queue empty
RETURN matches
```

* A dictionary is a list of *key* → *value* mappings, represented as {*key* = *value*, *key* = *value*, …}.

Figure 1. Pseudocode for ScrapRE search routine.

The following steps are performed for each source document (represented in memory as string *src* with *len* total characters indexed from *src*[0] through *src*[*len*-1]):

The routine scans for pattern matches efficiently and thoroughly by using flags to mark the last matched region for a particular pattern. This technique is important because there might be more than one occurrence of the pattern in a given source document, or the pattern might not occur at all.

The patterns are constructed in a manner such that a specific region in the source may actually meet the criteria of several patterns at once, but only the best match should be returned. Each time *flag* is reassigned, a search is performed for every pattern, starting from *flag*. ScrapRE considers whether *flag* is set in the middle of an already matched region; in such a case, to avoid redundancy, the search starts from the end of the matched region in which *flag* is set.

3. Documentation

Figures are included in this section that show examples from ScrapRE configuration files and demonstrative terminal sessions. The configuration files are intended to be customized but appropriate environmental adjustments must be made accordingly. Sessions are shown in a Linux terminal and the commands can be used in most Unix terminal environments, including Mac OS X. Windows installations do not include Python, so users may need to install the language packages and/or a development environment such as IDLE. Windows users may need to modify the commands given in this section.

3.1 System Requirements

ScrapRE was developed and tested on Red Hat Enterprise Linux with Python 2.4. The utility uses only standard Python modules and is compatible with Python 2.3 or later.

Sufficient permissions must be granted for directories to which ScrapRE will be writing output files. ScrapRE will automatically create missing directories as needed with sufficient permissions, but the utility may be unable to modify existing read-only directories. Permissions-related errors can be avoided by creating output directories (relative paths defined in *globals.conf*) before execution and manually changing the permissions as needed.

3.2 Setup and Configuration

No installer is included in this package. Unpack archived files from *ScrapRE.tar.gz* into *ScrapRE/* by issuing the terminal command:

```
> $ tar zxvf ScrapRE.tar.gz
```

ScrapRE/config/ contains two configuration files that may be edited to modify ScrapRE behavior; in both files, lines beginning with “#” are treated as comments and ignored. The global paths configuration file, *globals.conf*, shown in figure 2, contains four global path variables:

```
conf_links = config/links.conf
dir_wrappers = config/patterns/
dir_scraped = output/scraped/
dir_sources = output/sources/
```

Figure 2. Global paths configuration file, globals.conf (default).

The links configuration file, *links.conf*, shown in figure 3, must exist at the location specified for *conf_links*, and all wrappers must be contained in sub-directories of the path specified for *dir_wrappers*. The two output directories specified for *dir_scraped* and *dir_sources* will be created at runtime if they do not exist.

```
1 [bbc]
2    http://www.bbc.co.uk/urdu/index.xml
3
4 [voa]
5    http://www.voanews.com/urdu/...?keyword=Health
6    http://www.voanews.com/urdu/...?keyword=TopStories
7    http://www.voanews.com/urdu/...?keyword=Pakistan
8    http://www.voanews.com/urdu/...?keyword=South%20Asia
9    http://www.voanews.com/urdu/...?keyword=Politics
```

Figure 3. Links configuration file, links.conf (default, partial).

Each line in *links.conf* contains either a link or a domain label. Labels are single words nested between the “[”and “]” characters and derive from topmost Internet domain names. All non-empty, non-labeled lines are either article or RSS feed links, grouped with the closest preceding domain label.

For each domain labeled in *links.conf*, there exists a similarly labeled sub-directory in *dir_wrappers* that contains patterns to be searched for in a domain’s articles. For example, the article link on line 2 of *links.conf* belongs to the “bbc” domain group, labeled on line 1. ScrapRE will use patterns contained in *dir_wrappers/bbc/* to interface the article found at http://www.bbc.co.uk/urdu/index.xml.

Each *dir_wrappers* sub-directory contains an arbitrary number of text files defining regular expression patterns in conformance with Python’s verbose RE syntactical requirements. Each pattern interfaces a particular article element, and there may be multiple patterns for each document element (e.g., title, paragraph, image, or caption) being searched for in a domain, addressing variation from multiple templating schemes.

Pattern lines beginning with “!” name compilation flags defined in the `re` module;[*] valid flag names are DOTALL, IGNORECASE, and MULTILINE. All non-empty lines which do not contain compilation flags contain regular expressions. ScrapRE strips whitespace from and concatenates the RE lines, producing a pattern string to be compiled with the named flags (if any). Matches to the compiled patterns are searched for in every article from the associated domain groups. The *patterns/bbc/pattern_7 file*, shown in figure 4, is an example pattern for BBC Urdu articles.

```
1  !IGNORECASE
2  !MULTILINE
3  (<p class="storytext">)
4    ([(<!--.*-->)(<br>)]*)
5    (<table[^>]*>)
6      (<tr>)
7        ((<td[^>]*><img[^>]*></td>)?)
8        (<td><div><img src="([^"]*)"[^>]*></div></td>)
9      (</tr>)
10     (<tr>)
11       (<td class="caption">([^(</td>)]*)</td>)
12     (</tr>)
13   (</table>)
14   (?P<p>[^<>]*)
15   ((<br>)?[^<>]*)
16 (</p>)
```

Figure 4. Sample BBC Urdu article pattern file, patterns/bbc/pattern_7, with named paragraph entity (labeled “p”) highlighted.

Compilation flags IGNORECASE and MULTILINE are named in the first two lines of figure 4; lines 3-16 define a paragraph pattern. Named entity “p” is defined in line 14 as “[^<>]*,” a string of unspecified length not containing the characters “<” or “>”.

3.3 Usage

ScrapRE/scrape.py is the main script and can be executed with a call to the Python interpreter. ScrapRE behavior is determined by three run modes: *links*, *print*, and *verbose*. Each mode has a default setting[†] and an alternate setting, which can be invoked with command-line flags.

[*] See the Python Library Reference for `re` module documentation: http://docs.python.org/lib/module-re.html.
[†] Default run mode settings are denoted by superscript “D.”

- The *links* mode (articles or feeds^{D}) determines how URLs from *links.conf* are processed. If *links*=articles, only article webpage URLs are allowed; if *links*=feeds, URLs may only reference RSS feeds.[*] Also, *links.conf* may not contain both article and feed URLs.
- The *print* mode (true or false^{D}) determines whether scraped article entities are displayed in the console. If *print*=true, annotated entities are displayed;[†] if *print*=false, ScrapRE output is silenced.
- The *verbose* mode (true or false^{D}) determines whether non-critical system messages are displayed in the console. If *verbose*=true, ScrapRE activity reports are displayed; this options allows the user to track ScrapRE processes. If *verbose*=false, non-critical messages are silenced. Error messages resulting from execution failures are always displayed, regardless of the *verbose* setting.

To run ScrapRE with default settings for all modes (*links*=feeds, *print*=false, *verbose*=false), the user should issue the following command:

```
> $ python scrape.py
```

ScrapRE usage notes can be displayed by appending the -h or --help flag as follows:

```
> $ python scrape.py --help
> Usage: python scrape.py [option, ...]
>
> Options:
> --version      show program's version number and exit
> -h, --help     show this help message and exit
> -p, --print    show scraped text entities
> -v, --verbose  show non-critical runtime messages
> -l, --links    read URLs in links.conf as feed links (instead > of article links)
```

The -p, -v, and -f flags described in the last three lines above can be used in any combination to invoke alternate *print*, *verbose*, and *links* settings, respectively. For example, if *links.conf* contains article URLs instead of feed URLs, the alternate *links* setting must be flagged:

```
> $ python scrape.py -a
```

Alternate *print* and *verbose* settings must be flagged to display scraped article entities and non-critical system messages in the terminal:

```
> $ python scrape.py -pv
```

3.4 Output

ScrapRE outputs scraped text files to *ScrapRE/output/scraped/* and source files to *ScrapRE/output/sources/* (default behavior configured in *globals.conf*). The scraped files contain all matched entities with annotations, in order of occurrence within the source file.

[*]RSS feed URLs should have *.rss* or *.xml* extensions.

[†]The article entities displayed when *print* =True are persisted to *dir_scraped* as text files.

4. Conclusion

ScrapRE is a configurable Python module with a command-line user interface designed to be used interactively or as a scheduled daemon. Individual source articles can be specified directly with webpage links, or as batches with RSS feed links. Annotation syntax, input and output directories, and regular expression settings can all be configured by the user; such flexibility allows the ScrapRE to be easily ported across systems.

ScrapRE was developed to produce an annotated Urdu-language text corpora for ARL's ACUTE initiative. Corpora documents comprise training sets for machine translation systems, providing a valuable source of real semi-structured news data. The program has been used successfully for two years at the time of publication and has contributed significantly to ACUTE Urdu translation achievements.

NO. OF COPIES	ORGANIZATION
1 (PDF only)	DEFENSE TECHNICAL INFORMATION CTR DTIC OCA 8725 JOHN J KINGMAN RD STE 0944 FORT BELVOIR VA 22060-6218
1	DIRECTOR US ARMY RESEARCH LAB IMNE ALC HRR 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIM L 2800 POWDER MILL RD ADELPHI MD 20783-1197
1	DIRECTOR US ARMY RESEARCH LAB RDRL CIM P 2800 POWDER MILL RD ADELPHI MD 20783-1197

ABERDEEN PROVING GROUND

1	DIR USARL RDRL CIM G (BLDG 4600)

NO. OF COPIES	ORGANIZATION
1	DIRECTOR US ARMY RESEARCH LAB RDRL CII B BROOME 2800 POWDER MILL RD ADELPHI MD 20783-1197
2	DIRECTOR US ARMY RESEARCH LAB RDRL CII T C VOSS 2800 POWDER MILL RD ADELPHI MD 20783-1197
	ABERDEEN PROVING GROUND
11	DIR USARL RDRL CII C A BORNSTEIN (10 CPS) D WELSH

INTENTIONALLY LEFT BLANK.